LISTING OF THE CLAIMS

Please amend the claims as shown below. Please cancel Claims 4-5, 13 and 26 without prejudice. This Listing of the Claims replaces all prior versions and listings of claims in the Application.

1. (Amended) A computer implemented method for enforcing a consistent cacheability attribute for a page of physical memory, comprising:

storing a current cacheability characteristic attributed to a physical address of said page of physical memory;

accessing an identifier of said page of physical memory and an associated desired cacheability state corresponding to said physical memory page from a candidate entry to a translation lookaside buffer;

comparing said current cacheability characteristic and said desired cacheability state;

upon a match, entering said candidate entry to said translation lookaside buffer; and

upon a mismatch, generating an exception to enforce ~~wherein said exception causes enforcement of~~ said consistent cacheability characteristic wherein said current cacheability characteristic is conformed to said desired cacheability state with:

determining a cacheability transition to conform said current cacheability characteristic to said desired cacheability state;

determining a conservatism character of a mode of execution;

upon determining that said conservatism character is aggressive:

rolling back to the last commit point in said execution;

changing said conservatism character to conservative, wherein said changing comprises disallowing speculation; and

repeating said execution without speculation until said exception is reached;

upon determining that said cacheability transition comprises a cacheable to non-cacheable transition, performing said cacheable to non-cacheable transition; and

upon determining that said cacheability transition comprises a non-cacheable to cacheable transition, performing said non-cacheable to cacheable transition.

2. (Original) The method as recited in Claim 1 wherein said storing, accessing, and comparing are performed upon receiving said candidate entry to said translation lookaside buffer.

3. (Original) The method as recited in Claim 1 wherein said storing comprises mapping said physical address of said page of physical memory to a table wherein said table correlates said physical address to an entry indicating the current attributed cacheability characteristic of said page of physical memory.

4-5. (Cancelled)

6. (Amended) The method as recited in Claim 1 [[5]] wherein said performing said cacheable to non-cacheable transition comprises:

removing an old entry with said physical page identifier from said translation lookaside buffer;

updating said current cacheability characteristic to non-cacheable;

flushing a cache of said physical memory page; and

performing said entering.

7. (Amended) The method as recited in Claim 1 [[5]] wherein said performing said non-cacheable to cacheable transition comprises:

draining a pending write from a non-cacheable path;

removing an old entry with said physical page identifier from said translation lookaside buffer;

updating said current cacheability characteristic to cacheable; and

performing said entering.

8. (Original) The method as recited in Claim 1 wherein said accessing comprises:

walking a page table having a plurality of page table entries;

using a virtual address to find a page table entry corresponding to said physical memory page;

extracting said page table entry corresponding to said physical memory page; and

determining said physical page identifier and said desired cacheability state from said page table entry.

9. (Amended) A computer implemented method for entering a candidate entry to a translation lookaside buffer, comprising:

obtaining a first cacheability bit from said candidate entry;

obtaining a second cacheability bit from a table that associates a physical page identifier with a current attributed cacheability characteristic;

comparing said first cacheability bit and said second cacheability bit;

upon detecting a mismatch, generating an exception and performing a fix-up operation wherein said fix-up operation conforms said second cacheability bit with said first cacheability bit and comprises:

determining that a mode of execution is aggressive;

rolling back to the last commit point in said execution;

changing said mode of execution to disallow speculation; and

repeating said execution without speculation until said exception is triggered again; and

entering said candidate to said translation lookaside buffer.

10. (Original) The method as recited in Claim 9 wherein said physical page identifier comprises a physical page number.

11. (Original) The method as recited in Claim 9 wherein said first cacheability bit indicates cacheable and said second cacheability bit indicates non-cacheable and wherein said performing a fix-up operation comprises:

updating said current cacheability characteristic to non-cacheable; and

flushing a cache of said physical memory page.

12. (Original) The method as recited in Claim 9 wherein said first cacheability bit indicates non-cacheable and said second cacheability bit indicates cacheable and wherein said performing a fix-up operation comprises:

draining a pending write from a non-cacheable path; and

updating said current cacheability characteristic to cacheable.

13. (Cancelled)

14. (Amended) The method as recited in Claim 9 [[13]] wherein said first cacheability bit indicates cacheable and said second cacheability bit indicates non-cacheable and wherein said performing a fix-up operation comprises:

updating said current cacheability characteristic to non-cacheable; and

flushing a cache of said physical memory page.

15. (Amended) The method as recited in Claim 9 [[13]] wherein said first cacheability bit indicates non-cacheable and said second cacheability bit indicates cacheable and wherein said performing a fix-up operation comprises:

draining a pending write from a non-cacheable path; and

updating said current cacheability characteristic to cacheable.

16. (Original) The method as recited in Claim 9 further comprising removing an old entry with said physical page identifier from said translation lookaside buffer.

17. (Withdrawn) ~~A data structure disposed in a computer readable memory for providing information corresponding to the current cacheability characteristic attributed to a page of physical memory, said data structure comprising a data field for identifying said current cacheability characteristic, wherein said data structure is indexed by a physical page identifier.~~

18. (Withdrawn) ~~The data structure as recited in Claim 17 wherein said first data field comprises a page number corresponding to said page of physical memory.~~

19. (Withdrawn) ~~The data structure as recited in Claim 17 wherein said computer readable memory comprises a private memory.~~

20. (Amended) A computer based system for enforcing a consistent cacheability attribute for a page of physical memory upon entry of said page into a translation lookaside buffer, comprising:

a comparator for comparing an current cacheability characteristic attributed to said page of physical memory and a desired cacheability state for said page, wherein said comparator is coupled to said translation lookaside buffer;

a miss handler coupled to said comparator and to said translation lookaside buffer for handling a miss in said translation lookaside buffer upon said comparator determining that said current cacheability characteristic and said desired cacheability state match; and

an exception handler coupled to said comparator and to said translation lookaside buffer for handling an exception generated by said comparator upon said comparator determining that said current cacheability characteristic and said desired cacheability state do not match, wherein said handling conforms said second cacheability bit with said first cacheability bit and comprises:

determining that a mode of execution is aggressive;

rolling back to the last commit point in said execution;

changing said mode of execution to disallow speculation; and

repeating said execution without speculation until said exception is triggered again.

21. (Original) The system as recited in Claim 20 further comprising a page table entry walker-extractor coupled to said translation lookaside buffer for accessing a page table entry in a physical memory, walking said page table entry, locating an appropriate page table entry, and extracting said appropriate page table entry for said translation lookaside buffer.

22. (Original) The system as recited in Claim 20 wherein said system is deployed within a processor.

23. (Amended) A computer implemented method for maintaining cache coherency, comprising:

comparing a current cacheability characteristic attributed to a memory page physical address and a desired cacheability state for said address in a candidate entry to a translation lookaside buffer;

upon said comparing, detecting a mismatch between said current cacheability characteristic and said desired cacheability state;

upon said detecting, determining a cacheability mode transition to correct said mismatch; and

upon said determining, generating an exception;

upon said generating, correcting said mismatch wherein said correcting comprises:

upon said determining, ascertaining that a mode of execution is aggressive;

rolling back to the last commit point in said execution;

changing said mode of execution, wherein said changing comprises disallowing speculation; and

repeating said execution without speculation until said generating is triggered.

24. (Original) The method as recited in Claim 23 wherein said cacheability mode transition comprises a transition from cacheable to non-cacheable and wherein said correcting comprises:

removing an old entry with said address from said translation lookaside buffer;

updating said current cacheability characteristic to non-cacheable;

flushing a cache of a page corresponding to said address; and

entering said candidate entry to said translation lookaside buffer.

25. (Original) The method as recited in Claim 23 wherein said cacheability mode transition comprises a transition from non-cacheable to cacheable and wherein said correcting comprises:

draining a pending write from a non-cacheable path;

removing an old entry with said physical page identifier from said translation lookaside buffer;

updating said current cacheability characteristic to cacheable; and

entering said candidate entry to said translation lookaside buffer.

26. (Cancelled)

27. (Amended) The method as recited in Claim 23 [[26]] wherein said cacheability mode transition comprises a transition from cacheable to non-cacheable and wherein said correcting further comprises:

removing an old entry with said address from said translation lookaside buffer;

updating said current cacheability characteristic to non-cacheable;

flushing a cache of a page corresponding to said address; and

entering said candidate entry to said translation lookaside buffer.

28. (Amended) The method as recited in Claim 23 [[26]] wherein said cacheability mode transition comprises a transition from non-cacheable to cacheable and wherein said correcting further comprises:

draining a pending write from a non-cacheable path;

removing an old entry with said physical page identifier from said translation lookaside buffer;

updating said current cacheability characteristic to cacheable; and

entering said candidate entry to said translation lookaside buffer.